




# Lesson 1 – On Board LED

There is an LED on the Pico board, which is called the on-board LED. The onboard LED is connected to the GPIO25 pin. If you look closely at the Pico pins, you will see that GPIO25 is one of the hidden pins, which means that this pin is already occupied and other devices cannot use this pin. The advantage of this design is that the output can still be used to test the program even without connecting any external components. Next, let's test this LED together!

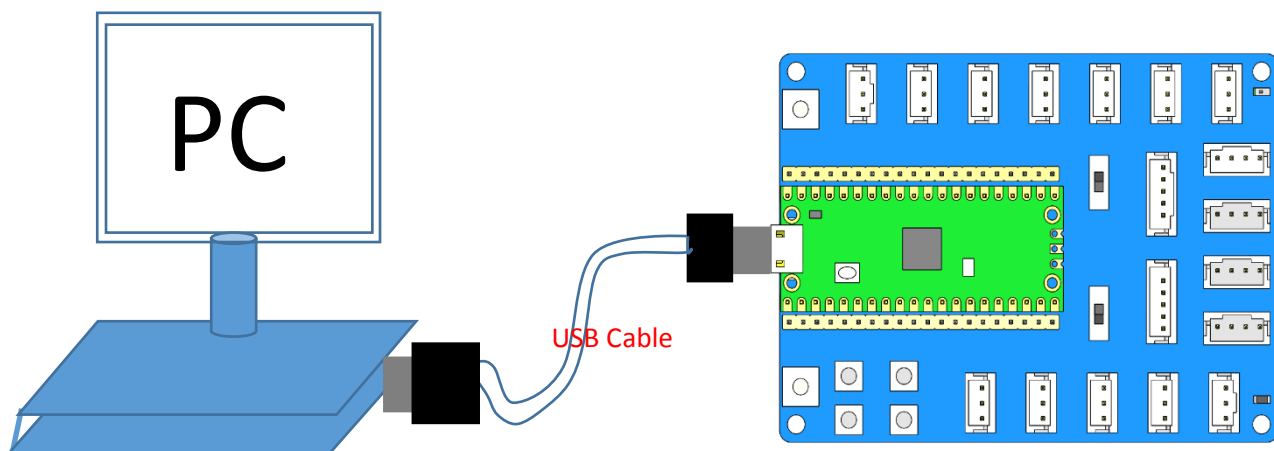
## 1.1 Component List

You need to prepare a Raspberry Pi Pico, a USB cable and a Pico expansion board

Component	QTY	Picture
Raspberry Pi Pico	1	
USB Cable	1	
PICO Expansion Board	1	

## 1.2 Circuit

Please refer to the picture below to install the Pico on the expansion board. Note that the USB port of the Pico should be in the same direction as the power port of the expansion board. After installation, connect the Pico to the computer via a USB cable.



## 1.3 Run Python Code

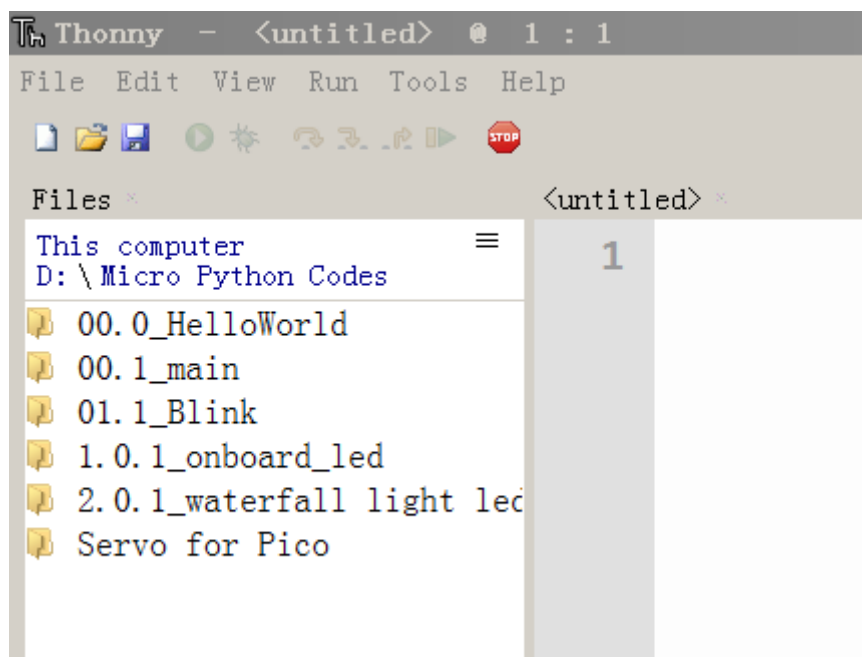
### Configuring the Operating Environment For Python

Please refer to the document “[2 Configuring the Operating Environment For Python](#)” which saved in [Pico\\_Extension\\_Board Kit Tutorial](#)”

1.3.1 Codes used in this tutorial are saved in [Pico\\_Extension\\_Board KitTutorial\Codes \Python\\_Codes](#). You can move the codes to any location. For example, we save the codes in Disk(D) with the path of [D:/ Micro Python codes](#).”

#### onboard\_led

1.3.2 Open “Thonny”, click “This computer”--“D:”--“Micro Python Codes”.



1.3.3 Expand folder “1.0.1\_onboard\_led” and double click “onboard\_led.py” to open it. As shown in the illustration below.

```

1. import machine
2. import utime
3.
4. # Define the GPIO port of the onboard LED as a GP:
5. led_onboard = machine.Pin(25, machine.Pin.OUT)
6.
7. while True:
8.     led_onboard.value(1)      # Output high level.
9.     print("Light on...")
10.    utime.sleep(2)             # Delay 2s.
11.    led_onboard.value(0)      # Output low level.
12.    print("Light off.")
13.    utime.sleep(2)             # Delay 2s
14.

```

Shell

Unable to connect to COM7: port not found

- 1.3.3 Make sure Raspberry Pi Pico has been connected with the computer. Click “Stop/Restart backend”, and then wait to see what interface will show up.

```

1. import machine
2. import utime
3.
4. # Define the GPIO port of the onboard LED as a GP25 p:
5. led_onboard = machine.Pin(25, machine.Pin.OUT)
6.
7. while True:
8.     led_onboard.value(1)      # Output high level.
9.     print("Light on...")
10.    utime.sleep(2)             # Delay 2s.
11.    led_onboard.value(0)      # Output low level.
12.    print("Light off.")
13.    utime.sleep(1)             # Delay 2s
14.

```

run current code

Stop/Restart backend

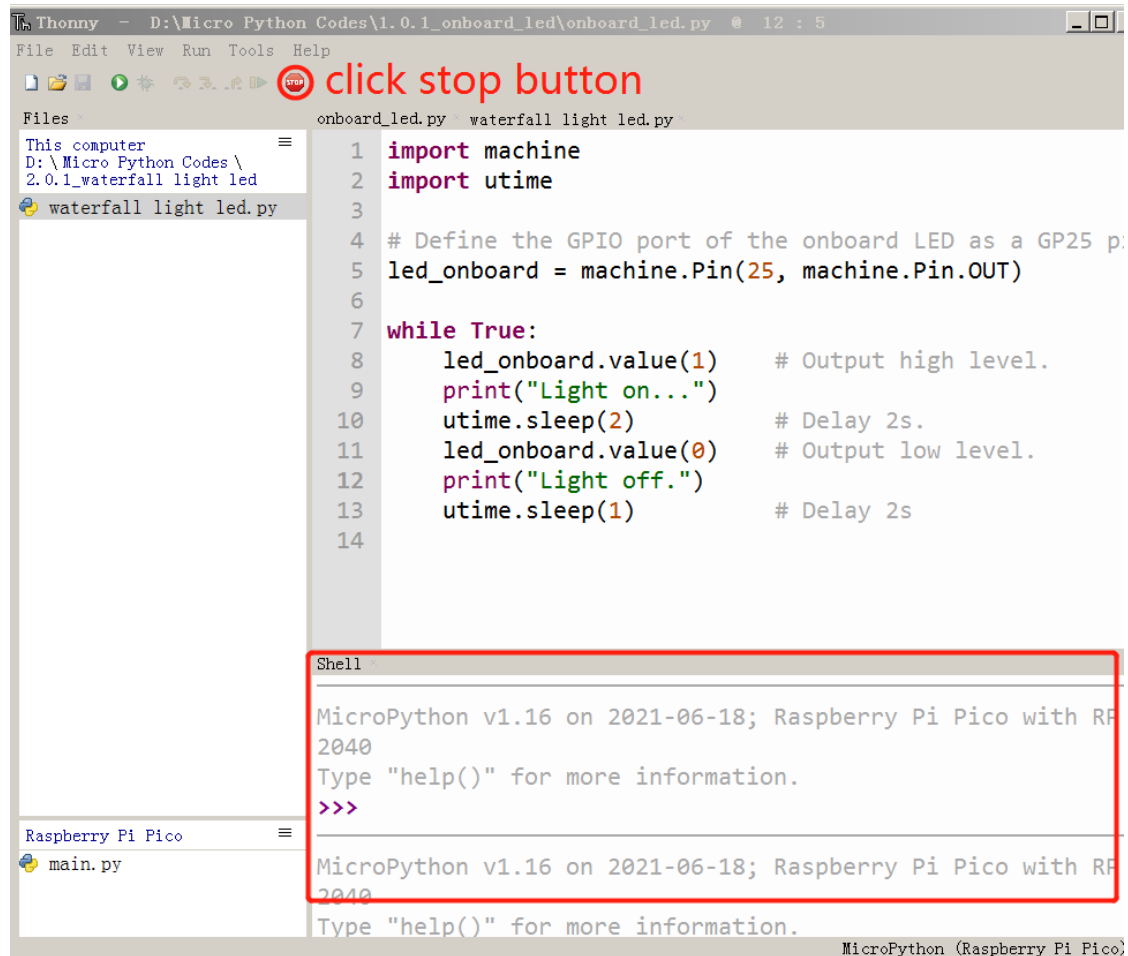
The information printed in the shell

Light off.  
Light on...  
Light off.  
Light on...  
Light off.  
Light on...  
Light off.  
Light on...  
Light off.  
Light on...

1.3.4 Click the run button, the current code will run and the LED on the Pico is blinking.

1.3.5 Click the stop button, the current code will stop and the LED on the Pico will turn off. The shell interface will no information print out.

**Note:** if the shell interface appears error message, please re-connect USB cable with Pico, or press the “RESET” button on Expansion Board to reset the Pico.



## 1.4 Python code

The Python code “onboard\_led.py” like below

### onboard\_led.py

```
1. import machine
2. import utime
3.
4. # Define the GPIO port of the onboard LED as a GP25 pin and set it to output mode.
5. led_onboard = machine.Pin(25, machine.Pin.OUT)
6.
7. while True:
8.     led_onboard.value(1)    # Output high level.
9.     print("Light on...")
10.    utime.sleep(2)          # Delay 2s.
11.    led_onboard.value(0)    # Output low level.
12.    print("Light off.")
13.    utime.sleep(2)          # Delay 2s
```

Note:

Using GPIO requires importing the machine library.

```
import machine
```

Like the machine library, the utime library is needed to handle time-related things, including the delay time we used in the code.

```
import utime
```

## 1.5 What's Next?

THANK YOU for participating in this learning experience!

If you find errors, omissions or you have suggestions and/or questions about the Tutorial or component contents of this Kit, please feel free to contact us: [cokoino@outlook.com](mailto:cokoino@outlook.com)

We will make every effort to make changes and correct errors as soon as feasibly possible and publish a revised version.

If you want to learn more about Arduino, Raspberry Pi, Smart Cars, Robotics and other interesting products in science and technology, please continue to visit our website. We will continue to launch fun, cost-effective, innovative and exciting products.

<http://cokoino.com/>

Thank you again for choosing Cokoino products.